IN THE CLAIMS:

1. (currently amended): A graphics system comprising:

one or more memories configured to receive and store graphics data, wherein each memory comprises on a single integrated chip,

one or more RAM memories configured to store the graphics data,

a level two cache memory connected to each RAM memory, and

a level one cache memory connected to each of the level two cache memories;

an array of registers configured to store status information, wherein the status information tracks and indicates accesses to the graphics data in the level one cache, wherein the status information further indicates whether the graphics data is modified or unmodified; and

a memory request processor connected to the memories and to the array of registers, wherein the memory request processor is operable to [[transfer]] write-back graphics data [[from]] stored in one of the level one cache memories that the status information indicates is modified to one of the corresponding level two cache memories [[according to the status information]] when an empty memory cycle occurs.

2. (original): The graphics system of claim 1, wherein the graphics data comprises samples.

3. (original): The graphics system of claim 1, wherein the graphics data comprises pixels.

4. (previously presented): The graphics system of claim 1, wherein each level one cache memory is divided into logical blocks, and wherein each register of status information corresponds to one logical block.

5. (previously presented): The graphics system of claim 4, wherein the status information comprises:

a least recently used (LRU) count, wherein the LRU count indicates which logical block in each level one cache memory has been least recently accessed; and

a dirty block bit, wherein the dirty block bit indicates which portions of the graphics data in each level one cache memory has been modified.

6. (original): The graphics system of claim 1, further comprising a request queue connected to the memory request processor, wherein the request queue comprises a first-in-first-out (FIFO) storage structure, wherein the request queue is configured to receive and buffer memory requests, and wherein the request queue is further configured to output the memory requests to the memory request processor in response to control signals from the memory request processor.

7. (previously presented): The graphics system of claim 6, wherein the array of registers is divided into two distinct sets, wherein one set of registers stores status information indicative of a current state of each level one cache, and wherein the second set of registers stores status information indicative of the current state of each level one cache plus the predicted results of one or more memory requests pending in the request queue.

8. (previously presented): The graphics system of claim 1, wherein each memory further comprises a shift register connected to each RAM, wherein each shift register is configured to receive and store portions of the graphics data from each RAM, and wherein each shift register is further configured to output graphics data serially in response to an external clock signal.

9. (original): The graphics system of claim 8, further comprising a display device, wherein the display device displays images according to the graphics data.

10. (previously presented): The graphics system of claim 1, wherein each memory further comprises an arithmetic logic unit (ALU) connected to the level one cache memory, wherein the ALU is configured to:

receive as one operand graphics data from a source external to the memory;
receive as a second operand graphics data stored in the level one cache;
arithmetically combine the two operands according to a function defined by an external control signal; and
store the results of the arithmetic combination in the level one cache.

11-32. (withdrawn)

33. (currently amended): A graphics system comprising:
one or more memories configured to receive and store graphics data, wherein each memory comprises on a single integrated chip,
one or more RAM memories configured to store the graphics data,
a level two cache memory connected to each RAM memory, [[and]]
a level one cache memory [[connected to each of the level two cache memories;]], and
a global data bus connecting the level one cache memory to each of the level two cache memories;
an array of registers configured to store status information, wherein the status information indicates whether the graphics data is modified or unmodified; and
a memory request processor connected to the memories and to the array of registers, wherein the memory request processor is operable to transfer graphics data from any level one cache memory to a corresponding level two cache memory if the graphics data in the level one cache memory is indicated to be modified, and if the global data bus has an empty memory cycle for the transfer.

34. (previously presented): The graphics system of claim 33, wherein the transfer of graphics data is prompted on demand.

35. (previously presented): The graphics system of claim 33, wherein the transfer of graphics data is periodic.

36. (previously presented): The graphics system of claim 33, wherein the graphics data comprises samples.

37. (previously presented): The graphics system of claim 33, wherein the graphics data comprises pixels.

38. (previously presented): The graphics system of claim 33, wherein each level one cache memory is divided into logical blocks, and wherein each register of status information corresponds to one logical block.

39. (previously presented): The graphics system of claim 33, wherein the memory request processor is further operable to transfer graphics data from any level one cache memory to a corresponding level two cache memory and at the same time to the RAM memory connected to the level two cache memory.

40. (currently amended): A graphics system comprising:

one or more memories configured to receive and store graphics data, wherein each memory comprises on a single integrated chip,

one or more RAM memories configured to store the graphics data,

a level two cache memory connected to each RAM memory, and

a level one cache memory connected to each of the level two cache memories;

an array of registers configured to store status information, wherein the status information indicates whether the graphics data is modified or unmodified; and

a memory request processor connected to the memories and to the array of registers, wherein the memory request processor is operable to transfer graphics data from any level one cache memory to a corresponding level two cache memory if the graphics data in the level one cache memory is indicated to be modified, and wherein the transfer occurs during a next [[and an]] empty memory cycle [[is detected]].

41. (new): A method for write-back of modified graphics data, the method comprising:

a) testing a dirty tag bit corresponding to a block of graphics data currently under examination in a level one cache, wherein the dirty tag bit indicates whether the data in the block is modified;

b) reading a dirty tag bit corresponding to a next block of graphics data in the level one cache, if the dirty tag bit indicates that the current block is not modified;

c) stalling until an empty memory cycle is detected, if the dirty tag bit indicates that the current block is modified;

d) commanding a memory request processor to write-back the current block of graphics data from the level one cache to a corresponding level two cache when the empty memory cycle occurs;

e) modifying the dirty tag bit corresponding to the current block of graphics data to indicate that the block is no longer modified and is available for future allocation; and

f) repeating steps a) through e) for the next block of graphics data in the level one cache.

42. (new): The method of claim 41, wherein the level two cache is configured as a "write-through" cache, and as the current block is written to the level two cache it is also written through to an associated DRAM memory bank connected to the level two cache.

43. (new): The graphics system of claim 33, wherein the transfer of graphics data occurs each time there is modified graphics data in the level one cache, and there is an empty cycle on the global data bus.